TM No. 861010

NAVAL UNDERWATER SYSTEMS CENTER
NEW LONDON LABORATORY
NEW LONDON, CONNECTICUT 06320

Technical Memorandum

VAX DATA PROCESSING PROGRAMS FOR THE HIGH FREQUENCY ACOUSTICS '85 EXPERIMENT

Date: 27 January 1986

Prepared by: Walter S. Hauck III
WALTER S. HAUCK III
ELECTRONICS ENGINEER

REFERENCE ONLY

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
27 JAN 1986	**Technical Memo**	**27-01-1986 to 27-01-1986**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
VAX Data Processing Programs for the High Frequency Acoustics '85 Experiment	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) **Walter Hauck III**	5d. PROJECT NUMBER **A67001; A65000; and B69025**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Underwater Systems Center,New London,CT,06320**	8. PERFORMING ORGANIZATION REPORT NUMBER **TM No. 861010**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
NUWC2015

14. ABSTRACT
This memorandum documents a set of signal processing programs, written in FORTRAN, and used during the High Frequency Acoustics '85 experiment, for at-sea acoustic data analysis. These programs have been subsequently expanded to support current analysis efforts on both a MicroVAX I and VAX 11/780 computers.

15. SUBJECT TERMS
signal processing programs; FORTRAN; High frequency acoustics; MicroVAX; VAX

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**28**	

TM No. 861010

NAVAL UNDERWATER SYSTEMS CENTER
NEW LONDON LABORATORY
NEW LONDON, CONNECTICUT 06320

Technical Memorandum

VAX DATA PROCESSING PROGRAMS FOR THE HIGH FREQUENCY ACOUSTICS '85 EXPERIMENT

Date: 27 January 1986

Prepared by: Walter S. Hauck III
WALTER S. HAUCK III
ELECTRONICS ENGINEER

PREFACE

This work was performed under NUSC Project No. A67001, High Frequency Acoustics, Principal Investigator, Dr. William Roderick, A65000, EVA Support for Shipboard Sonar, Principal Investigator, P. Herstein and B69025, Broadband Bearing Time Project, Principal Investigator, P. Herstein.

The author would like to thank J. M. Tattersall for the frequency sampling filter program, and J. Nordquist for her numerous helpful suggestions.

TABLE OF CONTENTS

SUMMARY

This memorandum documents a set of signal processing programs, written in FORTRAN, and used during the High Frequency Acoustics '85 experiment, for at-sea acoustic data analysis. These programs have been subsequently expanded to support current analysis efforts on both a MicroVAX I and VAX 11/780 computers.

INTRODUCTION

This memorandum is intended to serve as the reference document for a collection of data analysis/signal processing programs designed under the High Frequency Acoustics and Broadband Bearing Time programs. These programs are intended to provide a standard set of data manipulation programs which may be useful to a reasonably wide collection of users.

The document is broken into 3 sections. Section 1 provides an overview of the program organization, and briefly describes how and where data is to be processed. Section 2 is a more complete description of the individual programs, including the required program inputs, and mathematical model used to write each program. The final section discusses how to write a compatible program to fill a need not currently present. It is assumed throughout this memorandum that the reader is reasonably familiar with VAX file structure and the Digital Command Language. VAX commands and file designations appear in upper case letters.

SECTION I - OVERVIEW

1. Overview

A large set of FORTRAN programs were developed in support of the High Frequency '85 sea test. The programs were originally developed on a MicroVAX I computer, but are now also resident on the VAX 11/780 computer designated V331. All the programs are stored in the directory DRB0:[WALT2]. To access these programs, the user must first execute the command procedure ANALYZE. This procedure sets up a series of keywords which execute command procedures. The actual programs are stored in the directory [WALT2.PROC], and data must be located in [WALT2.DATA]. A flow chart of this architecture is shown in Figure 1.

As an example, assume we wish to obtain a time series plot of a record named TEST.DAT. From the user's directory, issue the command

```
$ @V331::DRB0:[WALT2]ANALYZE
```

and then

```
$ PLOT TEST
```

This will produce a plot of the values stored in [WALT2.DATA]TEST.DAT versus sample number. The plotting program assumes the user is issuing commands from a Tektronics 4010, 4014, or other compatible terminal.

A list of the available programs is shown in Appendix A. They include linear and logarithmic plots, spectral analysis, correlation, averaging and statistics. The list of available programs is also located in [WALT2]DESCRIP.TXT.

2. Reading and Writing Data

The programs in [WALT2.PROC] all look for files in [WALT2.DATA] with a ".DAT" file type. These files are a single column of values, written in REAL*4 format, using the subroutine D6KREAD. The companion subroutine WRITEOUT is used by each program to read the data. Both routines are located in the object library [WALT2.PROC]WALT.OLB, and listing appear at the end of this document. The FORTRAN calls are;

```
CALL D6KREAD (X,LU,N)
CALL WRITEOUT (X,LU,N)
```

where X is the data array of length N, and LU is a FORTRAN logical unit number. It is assumed a file name of the type [WALT2.DATA]*.DAT is assigned to the unit number LU. This is not the most efficient way to store the data; instead it is possible to view or edit the data using a text editor, a decided advantage in testing new programs or checking data quality.

It is important to note that all of the files in [WALT2], and its subdirectories are unprotected, allowing any user access to any program or data file. Care must be excerised when creating and deleting files.

Another cautionary note is needed. Because most programs create one output file for one input file, it is a simple matter to collect several hundred data files in a single terminal session. Please copy any files you wish to save to your own directory, and delete the remaining files.

3. File Naming

The majority of the programs are designed to operate on a set of files with names of the form PREFIX**.DAT. PREFIX is a set of characters and numbers, less than 64 characters long, and ** is the file number, from 01 to 100000. This naming convention allows for repetitive operations on many files, without the need to specify each file completely. In the following section, when a set of files is required, it is assumed they are named with the PREFIX**.DAT format, and that only the PREFIX and the total number of files need be input to the program.

As an example, it is desired to average 10 files named [WALT2.DATA]TEST01.DAT to TEST10.DAT. This is accomplished by the commands;

```
$ @[WALT2]ANALYZE              (this command need only be issued at
                                the start of a terminal session)

$ AVERAGE                      (the program prompts with)
  FILE NAME PREFIX:  TEST
  START FILE NUMBER: 1
  TOTAL NUMBER OF FILES TO BE PROCESSED: 10
  OUTPUT FILE NAME: AVETEST10
```

The result is written to [WALT2.DATA]AVETEST10.DAT.

SECTION II - PROGRAM DESCRIPTIONS

AVERAGE - Prompts:

```
FILE NAME PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
OUTPUT FILE NAME:
```

AVERAGE computes the average of a set of N files by

$$\bar{X} = \frac{1}{N} \sum_{n=1}^{N} X_n \qquad (1)$$

and writes the averaged result to the specified output file name. The input files must be of the same length. If a file whose values are all zero is included in the name set, the averaged result will be affected.

AVE2 - Prompts:

```
INPUT NUMBER OF FILES:
INPUT NAME OF FILE:
        .
        .                    (prompts N times)
        .
INPUT NAME OF FILE:
```

AVE2 computes the average of N files with unrelated file names. The complete name of each file, without the ".DAT" suffix, is required. The result is plotted directly to the user's terminal. No averaged data file is written to disk. AVE2 will accept files of varying length, but the averaged result is only correct up to the length of the shortest file.

CROSS - Prompts:

```
INPUT FILE 1:
INPUT FILE 2:
OUTPUT FILE NAME:
```

CROSS computes the real magnitude of the crosscorrelation of two real input files by [1]

$$\chi(m) = \sum_{n=1}^{N} x(n)\, y(n + m) \qquad (2)$$

Each of the input files is Fourier Transformed via the FFT program listed in Appendix B. The arrays are multiplied and the result inverse transformed to obtain the crosscorrelation function. Since the result of the IFFT is complex, the complex magnitude, allowing for both positive and negative values is computed. The function is then normalized by the product of the energies of each input sequence. Only the first half of the correlation, representing the lag times associated with the time delays formed from subtracting the arrivals times from the first input file minus the second input file are computed. The result is an output file of the same length as the input file. If the two input files are of unequal length, CROSS traps, i.e., issues a warning message, exits, and does not write the output file. Note if the input file prefixes are the same, CROSS will compute the autocorrelation of the input file.

CROSSALL Prompts:

```
FILE NAME 1 PREFIX:
FILE NAME 2 PREFIX:
OUTPUT FILE PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
FFT SIZE IN SAMPLES:
```

CROSSALL computes the crosscorrelation of two sets of files, in an identical fashion to CROSS. However, instead of naming two input files, two prefixes and the number of files to be processed is required. CROSSALL need not start with the first file in a set. The input files may be shorter than the FFT size, and are padded with zeros, so the output files are of a length equal to the FFT size. Several other programs besides CROSS have a corresponding CROSSALL program. In each case, the basic program requires a complete file specification, while the corresponding *ALL program requires prefixes and number of files.

CROSS2 and CROSS2ALL - Identical to CROSS and CROSSALL except both the lead and lag time differences are output to a file. The output file is twice the length of the input files.

CORRPLOT - Prompts:

```
INPUT FILE NAME:
```

CORRPLOT is a special purpose version of PLOT to be used with files generated from CROSS2 and CROSS2ALL. The second half of the sequence (lead times) is plotted prior to the first half of the output file (lags). The X axis shows delay samples, with the zero delay sample in the center of the plot. Please refer to the descriptions of PLOT and PLOTDB.

DELAY - Prompts:

```
INPUT FILE NAME:
TOTAL NUMBER OF FILES TO BE PROCESSED:
OUTPUT FILE PREFIX:
NUMBER OF SAMPLES TO SHIFT:
OUTPUT FILE LENGTH IN SAMPLES:
```

The DELAY program reads data from a file whose length is assumed to be long compared to the desired number of samples in the output file. DELAY breaks up this file into a set of smaller files, with the first sample in the output file set being the sample whose number is equal to the number of delay samples input. If the number of delay samples is zero, no shift takes place. The last file in the output set is the same length as the other files only if the length of the input file plus the number of delay samples is a n integer multiple of the output file size.

DESIGNFILTER - Prompts:

```
INPUT TIME DOMAIN OUTPUT FILE NAME:
INPUT FREQUENCY DOMAIN OUTPUT FILE NAME:
INPUT NUMBER OF POINTS IN IMPULSE RESPONSE:
INPUT FILTER BAND EDGES:
```

DESIGNFILTER generates the impulse response of a bandpass filter, to be used with the FILTER program. The impulse response is written to [WALT2.DATA]*.FILT where * is the ouput file name. The filter band edges are input in normalized frequency, as shown in Figure 2. The actual filter is a cosine-tapered bandpass filter designed by frequency sampling, as discussed in Tretter [2]. The original FORTRAN program was provided by J. M. Tattersall [3]. The trick in designing a filter is keeping the number of points in the impulse response small, to minimize execution time, and making the filter long enough to have the desired frequency response.

The program also asks if the impulse and frequency response of the filter should be plotted. Answer yes to the impulse response prompt to also plot the frequency response.

ENV - Prompts:

```
INPUT FILE 1:
INPUT FILE 2:
OUTPUT FILE NAME:
```

ENV calculates the envelope of two quadrature input files by

$$\left| x(n) \right| = \left(x_c(n)^2 + x_s(n)^2 \right)^{1/2} \qquad (3)$$

where the first input file is assumed to represent the sine component of the analytic signal, and the second input file, the cosine component. The signal is assumed to be complex, so that its quadrature components are

$$x(n) = x_c(n) + jx_s(n) \tag{4}$$

where $x_c(n)$ is the cosine component and $x_s(n)$ is the sine component.

ENV also prompts for a desired output. The choices are; 1) immediate plot of 20 LOG of the envelope values versus sample number, 2) immediate of the linear values versus sample number, or 3) output to named output file.

ENVALL - Prompts:

```
INPUT FILE 1 PREFIX:
INPUT FILE 2 PREFIX:
OUTPUT FILE PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
```

ENVALL computes the envelope from two sets of input files. The calculation is identical to ENV, but has no output options. Output is to named output files.

FILTER - Prompts:

```
FILE NAME PREFIX:
FILTER FILE NAME:
OUTPUT FILE NAME:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
```

FILTER convolves the impulse response of a filter with a set of data files using FFTs. The filter impulse response is assumed to have been generated with DESIGNFILTER and has a ".FILT" file type. FILTER corrects for the delay of the filter, and outputs to a set of files.

HILBERT - Prompts:

```
INPUT FILE NAME:
OUTPUT FILE NAME:
```

HILBERT compute the envelope of a real data file by [4]

$$\left| x(n) \right| = \mathrm{SQRT}\left(x(n) + j\hat{x}(n) \right)^2 \tag{5}$$

where $\hat{x}(n)$ is the Hilbert transform of x(n). The Hilbert Transform is computed via an FFT by the relation

$$F\hat{X}(f) = -jX(f)\ \mathrm{sqn}(f) \tag{6}$$

where X(f) is the FFT of x(n). Figure 3 is a diagram of the processing used to calculate the signal envelope.

HILBERTALL - Prompts:

```
INPUT FILE PREFIX:
OUTPUT FILE PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
```

HILBERTALL computes the envelopes of a set of input files, and writes the result to the named ouput file.

INVERSE - Prompts:

```
INPUT FILE NAME:
OUTPUT FILE NAME:
DATA RECORD LENGTH:
```

INVERSE corrects the values in a correlation file by a triangular weighting given by

$$x(n) = x(n) * \frac{N}{N - M} \qquad (7)$$

where N is the file length and M is the sample number. This weighting is intended to correct the peak of a correlation function, when that peak does not occur at the first sample point. The value of the correction decreases for peaks beyond half the record length, and can yield normalized correlation values of greater than 1. Note the last sample in the record is defined to be zero.

INVERSEALL - Prompts:

```
FILE NAME PREFIX:
OUTPUT FILE NAME PREFIX:
FILE RECORD LENGTH:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
```

INVERSE program for a set of data files.

PARTPLOT - Prompts:

```
INPUT FILE NAME:
STARTING SAMPLE NUMBER:
FINAL SAMPLE NUMBER:
```

PARTPLOT plots a portion of a data file between the input sample limits. Please refer to the discussion of PLOT.

PEAKPICK - Prompts:

```
FILE NAME PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
MINIMUM SEARCH SAMPLE NUMBER:
MAXIMUM SEARCH SAMPLE NUMBER:
```

PEAKPICK computes the mean and standard deviation of the largest absolute value from a set of data files. The minimum and maximum search samples allow only a setion of the input file set to be search. Both the minimum and maximum search samples are included in the search. PEAKPICK produces a plot on the user's terminal of peak value versus file number, with the mean and standard deviation in the plot heading. Please refer to the description of PLOT for options concerning the PEAKPICK output.

PEAKPICK2 - Prompts: same as PEAKPICK

PEAKPICK2 computes the same values as PEAKPICK, but does not produce the peak versus sample number plot. Instead, the mean and standard deviation of the peak are written to a file with the name [WALT2.DATA]*PREFIX*.STAT, where *PREFIX* is input. The number of files and minimum and maximum search samples are also written to this file. This version does not require the user to have a Tektronix-compatible terminal.

PHASE - Prompts:

```
INPUT FILE 1:
INPUT FILE 2:
OUTPUT FILE NAME:
```

PHASE calculates the phase of 2 quadrature files by

$$\phi(n) = \tan^{-1}\left(\frac{x_s(n)}{x_c(n)}\right) \tag{8}$$

where the first file name represents $x_c(n)$, and the second file, $x_s(n)$. PHASE also asks the user if a plot is desired, or output to a file.

PLOT - Prompts:

```
INPUT FILE NAME:
```

PLOT produces a plot of data file values on the Y axis versus sample number. PLOT is a stand-alone version of a program called APLOT, a GRAFX-based interactive plotting routine [5]. The Y axis always spans the minimum and

maximum values in the data, but can be modified as described in Appendix C. Note: to change a plot parameter, press return and then a character followed by 2 returns; for help, press return, H and return. To end a plot press return twice.

PLOTDB - Prompts:

```
INPUT FILE NAME:
```

PLOTDB produces an XY plot, with the Y values 20*LOG of the data file values. All the PLOT options apply.

PSDALL - Prompts:

```
FILE 1 NAME PREFIX:
OUTPUT NAME PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
SAMPLING RATE IN HERTZ:
```

PSDALL computes the 10*LOG of the power spectral density, in dB/Hz, for a set of input files. The output file length is the next greatest multiple of 2**N, unless the file length is M is identically M=2**N. The power spectral density is the FFT of the input file, with the FFT coefficients divided by the record length in seconds.

QCROSS - Prompts:

```
INPUT COS FILE 1:
INPUT SIN FILE 1:
INPUT COS FILE 2:
INPUT SIN FILE 2:
OUTPUT FILE NAME:
```

Calculates the crosscorrelation of 2 sets of quadrature files. The signal are placed in complex arrays, as described in the PHASE program, and the complex magnitude of the correlation function output. As with CROSS, only the lag times are output, so the output file is the same length as the four input files.

QUAD - Prompts:

```
INPUT SIN FILE:
INPUT COS FILE:
OUTPUT FILE NAME:
```

QUAD computes 20*LOG of the FFT of 2 quadrature input files. See PHASE description for definition of quadrature files.

SAMPLE - Prompts:

```
FILE NAME PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
OUTPUT FILE NAME:
SAMPLE NUMBER OF INTEREST:
```

SAMPLE copies the value of the desired sample number from a set of files, and places these numbers sequentially in the output file. As an example, SAMPLE will copy the tenth value from file one into the first sample of the output file, the tenth value from file two into the second sample in the output file, etc., for each file in the data set.

SPTRM - Prompts:

```
INPUT FILE NAME:
OUTPUT FILE NAME:
```

SPTRM compute 20*log of the FFT of the input data file name. Output file is nearest factor of 2 to input data file length.

SPTRMALL - Prompts:

```
INPUT FILE PREFIX:
OUTPUT FILE PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
```

Same calculation as SPTRM, for a set of data files.

WATER - Prompts:

```
FILE NAME PREFIX:
STARTING FILE NUMBER:
TOTAL NUMBER OF FILES TO BE PROCESSED:
SCALE FACTOR:
SAMPLING FREQUENCY:
TIME MIN (MS):
TIME MAX (MS):
CENTER FOR DASHED LINE:
```

WATER produces a waterfall plot of a set of data files. The first data file is plotted at the top of the figure. Data may be vertically scaled by the SCALE FACTOR, and windowed by TIME MIN and MAX. Horizontal inputs are not in sample numbers, but time. A vertical dashed line may be drawn on the plot for non-zero times. Plot is not a PLOT output, and the plot must be ended with a control-Y.

WATER2 - Prompts: same as WATER

Special purpose version of WATER. Does not produce a plot to the user's terminal, but creates an IPF file in the current default directory. IPF file may be plotted using the Grafx Post Processor, GPP.

SECTION III - WRITING COMPATIBLE PROGRAMS

1. Subroutines

The subroutines D6KREAD, WRITEOUT and FFT are located in [WALT2.PROC]WALT.OLB. APLOT routines may be found in [HAUCK.APLOT]APLOT.OLB. The link statement is then

```
$ LINK PROGRAM,DRB0:[WALT2.PROC]WALT/LIB,
  [HAUCK.APLOT]APLOT/LIB,DBA0:[GRAFX]GRAFX/LIB  .
```

Any FORTRAN program is compatible if it uses D6KREAD and WRITEOUT, and input/outputs REAL*4 arrays.

2. Command Programs

All the programs described in this memorandum are a pair of programs; a FORTRAN to compute the quantity of interest and a DCL command procedure to handle file naming, and passing variables to the FORTRAN program. The shell of all the command procedures is either designed for a single file, or a set of files.

A example shell command procedure that acts on a single file is HILBERT.COM

```
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NAME:"
$ ASSIGN/USERMODE  [WALT2.DATA]'P1'.DAT FOR009
$ ASSIGN/USERMODE [WALT2.DATA]'P2'.DAT FOR011
$!
$ MOVEON:
$ ASSI/USERMODE  SYS$COMMAND FOR005
$ ASSI/USERMODE  SYS$COMMAND FOR$ACCEPT
$!
$ R [WALT2.PROC]HILBERT
$!
$EXIT
```

This procedure simply asks for two file names, assigns them to the correct FORTRAN units and executes the program. It does however free then user form executing the program from a specific directory, places the output in a particular directory, and does not require the user to input a complete file specification.

HILBERTALL illustrates how sequential file naming is handled;

```
$ IF P1 .EQS. "" THEN INQUIRE P1 "FILE 1 NAME PREFIX"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT NAME PREFIX"
$ PREFIX1=P1
$ PREFIX3=P2
$ IF P3 .EQS. "" THEN INQUIRE P3 "STARTING FILE NUMBER"
$ ISTART=F$INTEGER(P3)
$ IF P4 .EQS. "" THEN INQUIRE P4-
 "TOTAL NUMBER OF FILES TO BE PROCESSED"
$ NFILES=F$INTEGER(P4)
$    NCOUNT=1
$ PROCFILE:
$ IF NCOUNT.GT.NFILES THEN GOTO DONE
$!
$    FILENO=F$STRING(ISTART)
$    IF ISTART.LT.10 THEN FILENO="0"+FILENO
$    RAWFILE1="[WALT2.DATA]"+PREFIX1 + FILENO+".DAT"
$    OUTFILE1="[WALT2.DATA]"+PREFIX3 + FILENO+".DAT"
$    ASSIGN/USERMODE 'RAWFILE1' FOR009
$    ASSIGN/USERMODE 'OUTFILE1' FOR011
$    ASSIGN/USERMODE TEMPDAT.TMP FOR012
$!                     PROCESS EACH FILE
$ R [WALT2.PROC]HILBERT
$!
$    ISTART=ISTART+1
$    NCOUNT=NCOUNT+1
$    GOTO PROCFILE
$ DONE:
$EXIT
```

REFERENCES

1. A. Oppenheim and R. Schafer, DIGITAL SIGNAL PROCESSING, Prentice-Hall, Englewood Cliffs, NJ, 1975, pp. 559.

2. S. Tretter, DISCRETE-TIME SIGNAL PROCESSING, John Wiley and Sons, New York, 1976, pp. 235.

3. J. M. Tattersall, personal communication.

4. A. Whalen, DETECTION OF SIGNALS IN NOISE, Academic Press, New York, 1971, pp. 62.

5. W. S. Hauck and J. M. Monti, NUSC SER: 4333-25

APPENDIX A - AVAILABLE PROCESSING PROGRAMS

AVERAGE	average of a set of files
AVE2	average data file with unrelated names
CROSS	crosscorrelation of 2 files - lag times only
CROSSALL	crosscorrelation of 2 sets of files - lags only
CROSS2	crosscorrelation of 2 files - leads and lags
CROSS2ALL	crosscorrelation of 2 sets of files - leads and lags
CORRPLOT	plotting program for files created with CROSS2
DELAY	delays and breaks up a large data file
DESIGNFILTER	designs a filter. input frequencies in f/fs
ENV	envelope of 2 quadrature files
ENVALL	envelope of 2 sets of quadrature files
FILTER	filters a set of data files
HILBERT	calculates envelope of data file via hilbert transform
HILBERTALL	envelope via hilbert transform for a set of data files
INVERSE	corrects for triangular window of correlation file
INVERSEALL	corrects triangular window for a set of files
PARTPLOT	plots a section of a data file
PEAKPICK	statistics and plot of absolute peak value in window
PEAKPICK2	statistics of absolute peak output to *.stat file
PHASE	phase of 2 quadrature files
PLOT	data file plot
PLOTDB	20log plot of data file
PSDALL	10log power spectral density of set of data files
QCROSS	crosscorrelation of 2 sets of quadrature files
QUAD	20log spectrum of 2 quadrature files
SAMPLE	subsample a data file
SPTRM	20log spectrum of data file
SPTRMALL	20log spectrum of a set of data files
WATER	waterfall plot of a set of data files
WATER2	IPF file output for waterfall plot

APPENDIX B - SUBROUTINE LISTINGs

D6KREAD.FOR

```
      SUBROUTINE D6KREAD (X,LU,ALEN)
C
      REAL X(100000),Y(100000)
      INTEGER ALEN,LU
C
      ALEN=0
      DO 10 I=1,100000
      READ (LU,*,ERR=11) X(I)
      ALEN=ALEN+1
   10     CONTINUE
C                                        EXIT READ LOOP
   11     CONTINUE
C
      RETURN
      END
```

WRITEOUT.FOR

```
      SUBROUTINE WRITEOUT (X,LU,ILEN)
C
      REAL X(2000),Y(2000)
      INTEGER ILEN,LU
C
      DO 10 I=1,ILEN
      WRITE (LU,*) X(I)
   10     CONTINUE
C
      RETURN
      END
```

FFT.FOR

```
      SUBROUTINE FFT (X,N,INV)
C
C     X => COMPLEX ARRAY N POINTS IN LENGTH
C
C     N => TRANSFORM LENGTH
C
C     INV => 0 = FORWARD TRANSFORM
C            1 = INVERSE TRANSFORM
C
      COMPLEX X(1),U,W,T,CMPLX
C
      M=ALOG(FLOAT(N))/ALOG(2.0)+.1
      NV2=N/2
      NM1=N-1
      J=1
      DO 40 I=1,NM1
      IF (I.GE.J) GO TO 10
      T=X(J)
      X(J)=X(I)
      X(I)=T
   10 K=NV2
   20 IF (K.GE.J) GO TO 30
      J=J-K
      K=K/2
      GO TO 20
   30 J=J+K
   40 CONTINUE
      PI=4.0*ATAN(1.0)
      DO 70 L=1,M
      LE=2**L
      LE1=LE/2
      U=(1.0,0.0)
      W=CMPLX(COS(PI/FLOAT(LE1)),-SIN(PI/FLOAT(LE1)))
      IF (INV.NE.0) W=CONJG(W)
      DO 60 J=1,LE1
      DO 50 I=J,N,LE
      IP=I+LE1
      T=X(IP)*U
      X(IP)=X(I)-T
      X(I)=X(I)+T
   50 CONTINUE
      U=U*W
   60 CONTINUE
   70 CONTINUE
      IF (INV.EQ.0) RETURN
      DO 80 I=1,N
      X(I)=X(I)/CMPLX(FLOAT(N),0.0)
   80 CONTINUE
      RETURN
      END
```

APPENDIX C - DESCRIPTION OF PLOT PROGRAM

The PLOT program is a GRAFX-based subroutine which allows the user to interactively modify the characteristics of a plot on a Tektronix 4010/4014 compatible terminal. The default plot has axis value which span the data on both the X and Y axes. The FORTRAN call to APLOT is

```
CALL APLOT (X,Y,N,XAXIS,YAXIS,TITLE)
```

where X and Y are real arrays of length N, and XAXIS, YAXIS, and TITLE are CHARACTER*60 variables. PLOT has a help facility which may be invoked by pressing a carriage return, followed by an "H" and a second carriage return. PLOT responds with;

PLOT OPTIONS

PRESS	TO CHANGE
X	X AXIS
Y	Y AXIS
D	DIVISIONS ON AN AXIS
P	DECIMAL PLACES ON AN AXIS
T	TIC LINES IN X OR Y
M	MULTIPLE TIC LINES IN X OR Y
C	SINGLE LINE TITLE TO TWO LINE TITLE
I	PLOT TO INTERMEDIATE PLOT FILE (IPF)
I1	PLOT TO BOTH TERMINAL AND IPF
S	LABEL DATA POINTS WITH CIRCLES
Q	QUIT

HIT RETURN TO EXIT HELP AND RE-DISPLAY PLOT
TYPE A VALID LETTER FOR DESIRED PLOT OPTION

By pressing an "X" and return, PLOT responds with

```
INPUT XMIN,XMAX
```

The user enters the desired X axis values, separated by a space, and presses carriage return. PLOT responds with

```
ANOTHER CHANGE ?
```

If the userpresses return again, the plot is re-drawn with the new X axis values.

Multiple changes to a plot may be made without re-plotting each time. When PLOT asks for another change, respond with another valid letter, say "Y", and PLOT prompts

INPUT YMIN,YMAX

Enter the new Y axis limits and press two carriage returns. Now both the X and Y axis limits have been changed. Note "H" may be hit at any break point without effecting the current plotting parameters.

Viewing a plot is terminated by pressing two carriage returns without a valid letter. The APLOT subroutine is exited, and control returns to the calling program. Reference [5] is a more complete discussion of APLOT, but describes an earlier version named ALLPLOT. The two program are functionally identical, with APLOT having some additional features.

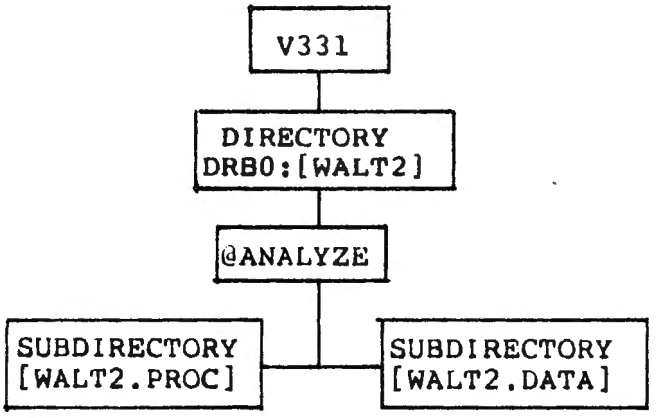

FIGURE 1 - VAX Processing Program Directories.

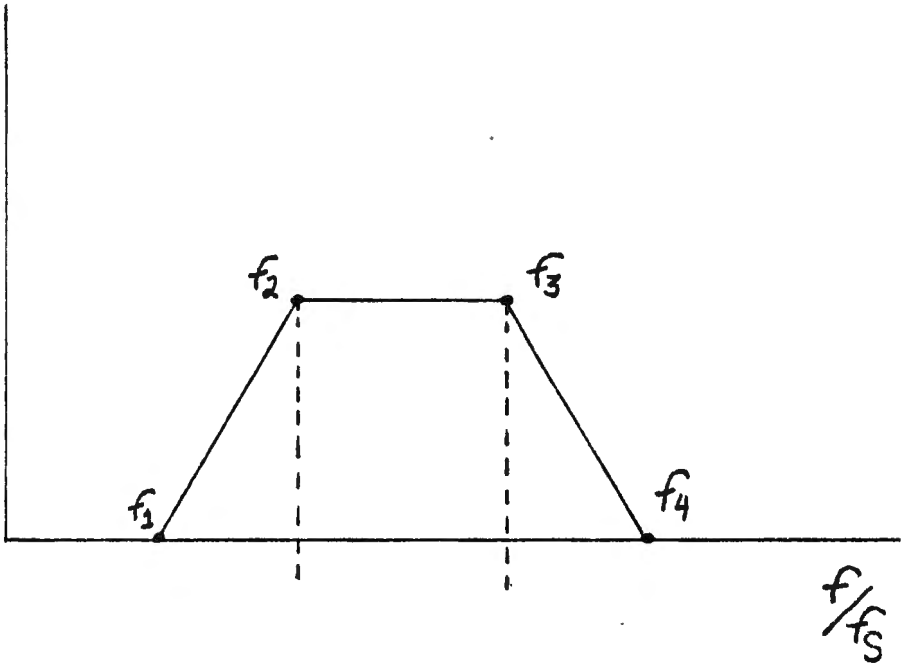

FIGURE 2 - DESIGNFILTER Program Input Frequency Definitions.

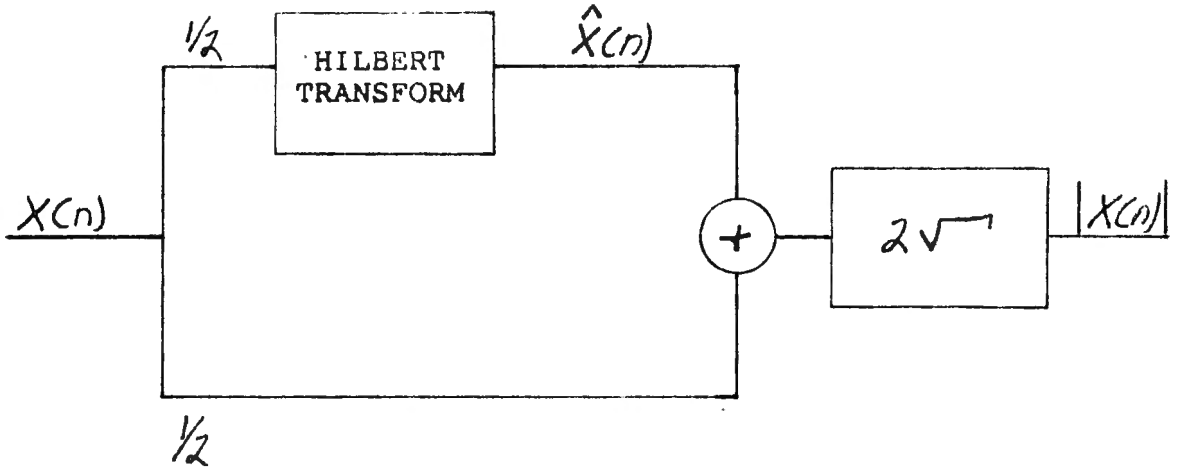

FIGURE 3 - Calculation of Signal Envelope via Hilbert Transform.

INTERNAL DISTRIBUTION

Code 33	L. Freeman
33A	B. Cole, W. Roderick
33A3	P. Herstein, M. Hallisey
331	R. Boivin
3314	G. Carter, D. Sheldon, K. Briotti
332	R. Radlinski
3322	R. Charette
3323	D. Bostian, L. Maiocco
333	W. Schumacher
333s	
3331	D. Browning, J. Chester, R. Dullea, P. Koenigs, B. Nuetzel J. Tattersall, J. Monti, R. Christan, G. Sharman, W. Hauck (S)
3332	H. Weinberg, R. Deavenport, H. Sternberg, G. Botseas, J. Nordquist, L. Petitpas, H. Tran
3212	W. Zwolinski
021311	(NLON) 3 copies
021312	(NPT) 3 copies

INTERNAL DISTRIBUTION

Code 33	L. Freeman
33A	B. Cole, W. Roderick
33A3	P. Herstein, M. Hallisey
331	R. Boivin
3314	G. Carter, D. Sheldon, K. Briotti
332	R. Radlinski
3322	R. Charette
3323	D. Bostian, L. Maiocco
333	W. Schumacher
333s	
3331	D. Browning, J. Chester, R. Dullea, P. Koenigs, B. Nuetzel J. Tattersall, J. Monti, R. Christan, G. Sharman, W. Hauck (S)
3332	H. Weinberg, R. Deavenport, H. Sternberg, G. Botseas, J. Nordquist, L. Petitpas, H. Tran
3212	W. Zwolinski
021311	(NLON) 3 copies
021312	(NPT) 3 copies